Quantum Program Generation Must Prioritize Validity Over Probabilistic Scaling

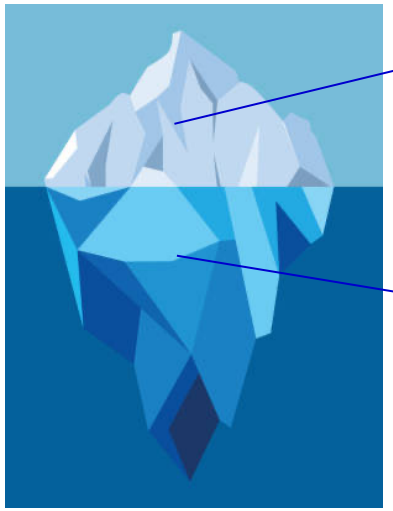
Junhao Song[1], Yu Zhou[1], William Knottenbelt[1], Yudong Cao[2,3]

1. Imperial College London **2.** BCG X AI Science Institute **3.** Zapata Quantum

ICML 2026. Seoul, South Korea. July 6th - 11th, 2026

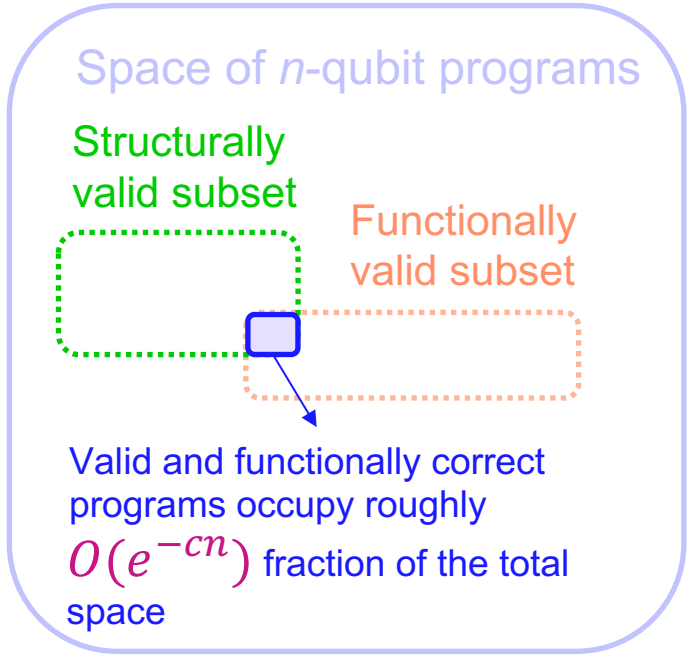
Our position

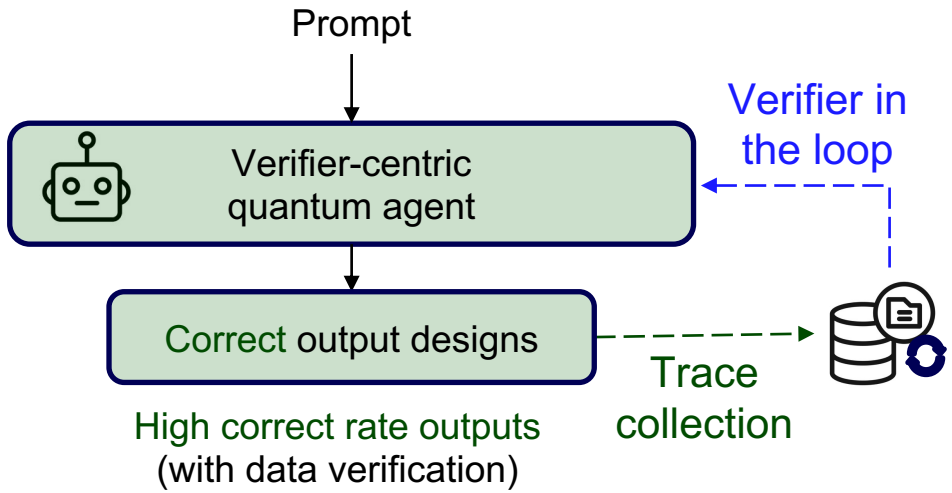
We argue that quantum program generation and other similar scientific domains with hard physical and mathematical constraints expose a fundamental syntax-semantics gap, where applying probabilistic scaling of large models can be counterproductive. The exponential sparsity of functionally correct outputs makes post-selection intractable, so reliability must come from a constructively verified system.



Syntax
Parsable IR, type checking

Semantics
Quantum-related constraints e.g. reversibility, ancilla uncomputation





BCG X | BCG X AI Science Institute | ZAPATA // QUANTUM | ICML International Conference On Machine Learning

Why quantum program generation is not "just another code task"

Comparison between classical and quantum

Syntax-semantics gap. The distance between syntax i.e. a program being well-formed and semantics i.e. it computing what was intended.

Classical programming:

- Gap exists, but is narrow in practice
- Cheap semantic checks: execution, tests, and debug
- Errors are localized

In quantum programs, syntactic fluency no longer tracks physical constraints of the computing device.

Classical Programming

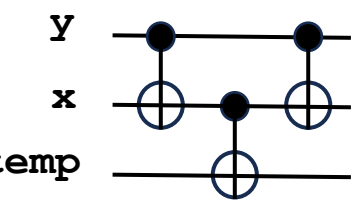
```
Line 1: x = y;
Line 2: temp = x;
Later program does not use `temp` variable
```

Ignore Line 2
Do not affect semantics

Syntax: valid
Semantics: valid

Quantum Programming

```
Line 1 => CNOT(y,x)
Line 2 => CNOT(x,temp)
Uncompute Line 1 => CNOT(y,x)
Ignore uncomputing Line 2
```



Ignore Line 2
Qubit y entangled with temp

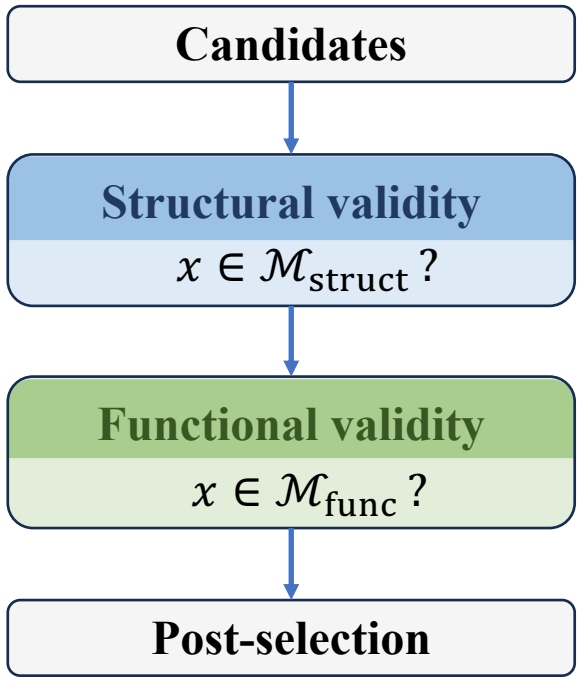
Syntax: valid
Semantics: invalid

BCG X | BCG X AI Science Institute | ZAPATA // QUANTUM | ICML International Conference On Machine Learning

Why “generate-then-filter” hits a wall

The exponential cost of post-hoc verification

Generate-then-filter. A pipeline where a model proposes *K* candidate programs from its learned distribution, and a downstream verifier discards those that violate the target constraints.



Probabilities of passing the filters for NISQ/FTQC regimes:

	$p_{\text{struct}} = Pr(x \in \mathcal{M}_{\text{struct}})$	$p_{\text{func}\|\text{struct}} = Pr(x \in \mathcal{M}_{\text{func}} \| x \in \mathcal{M}_{\text{struct}})$
Noisy Intermediate Scale Quantum (NISQ) devices	$\Theta(1)$	$O(e^{-\gamma_{\text{NISQ}} n})$
Fault Tolerant Quantum Computing (FTQC) devices	$O(e^{-\beta n})$	$O(e^{-\gamma_{\text{FTQC}} n})$

Cost of verification:

$x \in \mathcal{M}_{\text{struct}}$?	$x \in \mathcal{M}_{\text{func}} \| x \in \mathcal{M}_{\text{struct}}$?
$C_{\text{struct}} = O(nd)$	$C_{\text{func}\|\text{struct}} = O(2^n)$

Expected cost* of finding a functionally correct candidate

$$\mathbb{E}[C] = \frac{C_{\text{struct}}}{p_{\text{struct}} \cdot p_{\text{func}|\text{struct}}} + \frac{C_{\text{func}}}{p_{\text{func}|\text{struct}}} \in O(e^{cn}),$$ an **exponential** scaling.

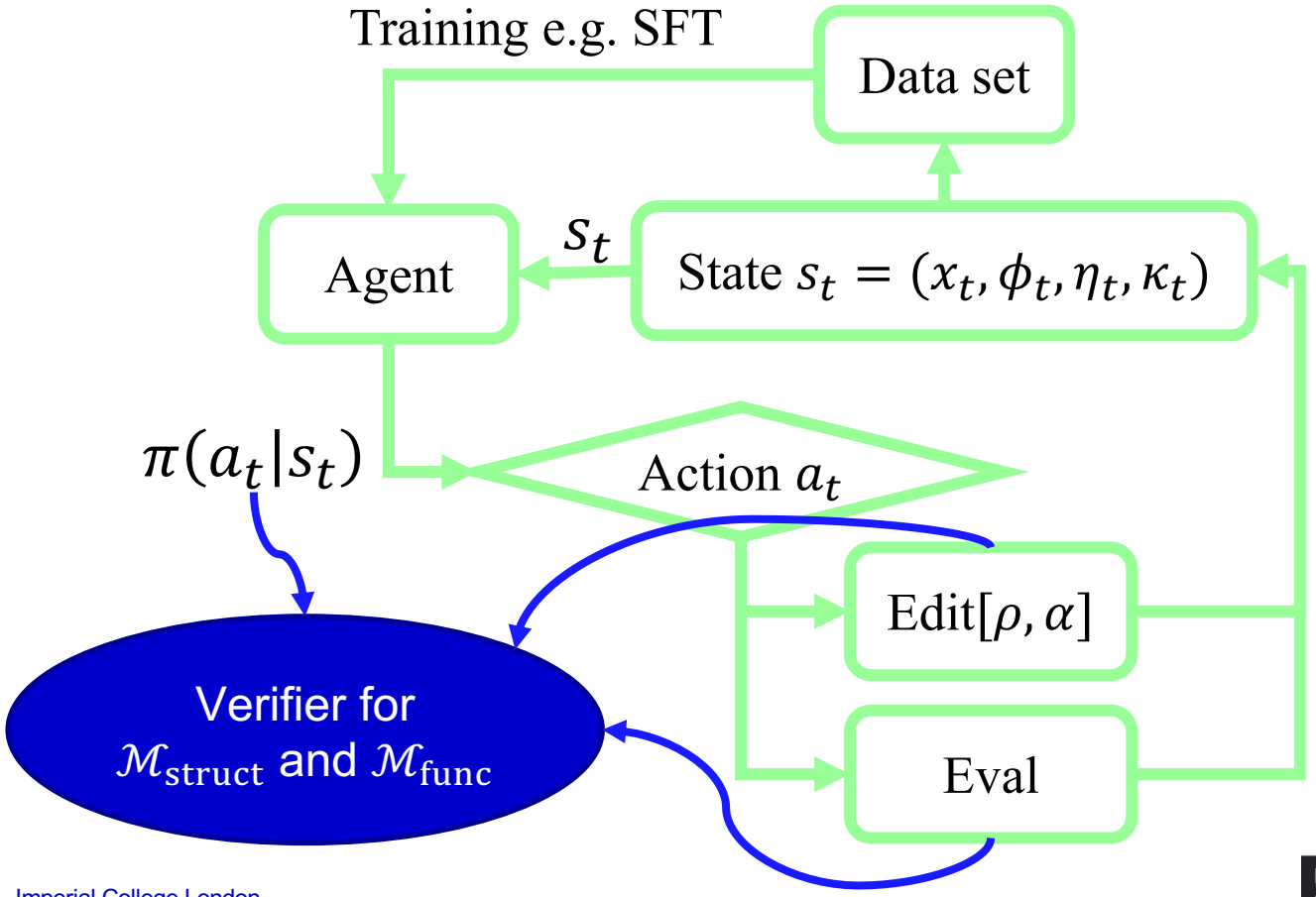
*Assuming poly-time samplers without target-specific memorization. See Appendix B and C for details.

BCG X | BCG X AI Science Institute | ZAPATA // QUANTUM | ICML International Conference On Machine Learning

From filters to invariants

Formulation of the verifier-centric agent architecture

Constructive verification. An design paradigm where the verifier implements constraints inside a generation loop.



Markov Decision Process (MDP)

State s_t at time t

- x_t: current program design
- ϕ_t: parameters
- η_t: metrics
- κ_t: auxiliary bookkeeping

Action a_t at time t

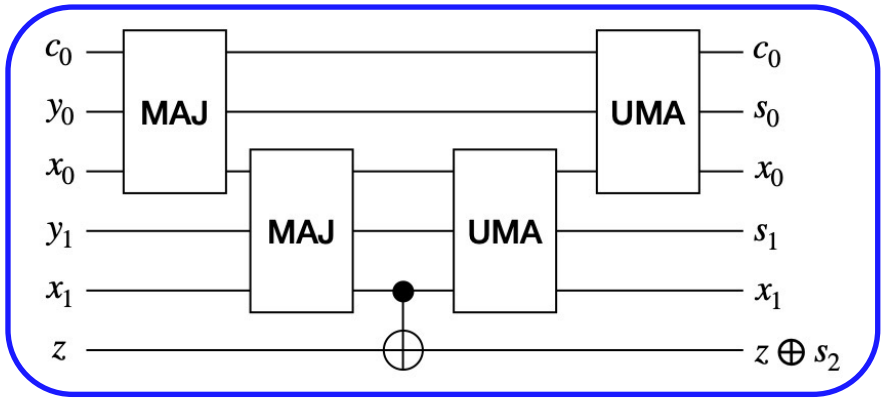
- $\mathrm{EDIT}[\rho, \alpha]$: apply property-preserving rewrite ρ with arguments α
- EVAL: evaluates metrics

BCG X | BCG X AI Science Institute | ZAPATA // QUANTUM | ICML International Conference On Machine Learning

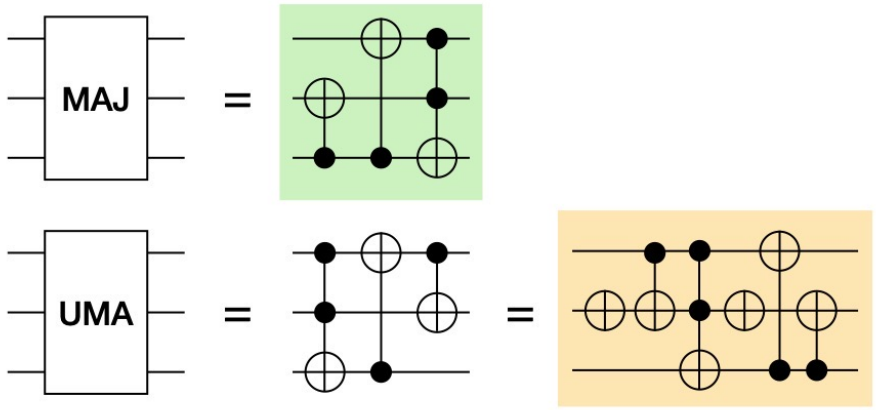
A rollout of the verifier-centric agent

Using the ripple-carry adder as an example

(a) Level 1: module based circuit design

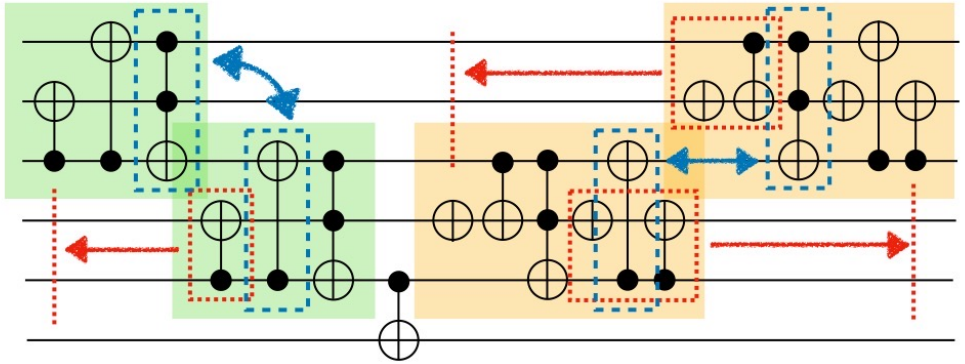


(b) Level 2: decomposition of individual modules



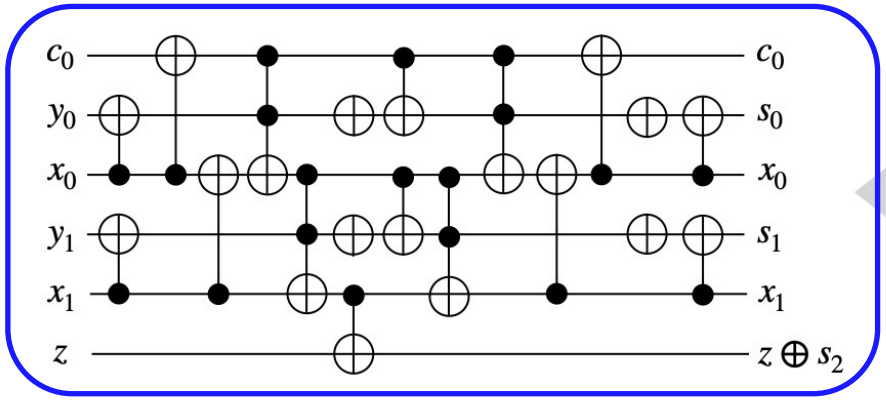
Program design x

(c) Level 3: property preserving rewrites $\mathrm{EDIT}[\rho, \alpha]$



Potential improvements based on precise mathematical insights about quantum circuits: parallelized execution, gate position swap

(d) Optimized circuit



Applying the improvements leads to an optimized circuit with significantly lower depth

Program design x'

Thank you for listening!